PETUNJUK PRAKTIKUM

PENGENALAN POLA

Disusun oleh :
Dr. Abdul Fadlil , M.T.
Ahmad Azhari, S.Kom., M.Eng.

Program Studi Teknik Informatika
Fakultas Teknologi Industri
Universitas Ahmad Dahlan
2019

KATA PENGANTAR

Puji syukur kami panjatkan kehadirat Allah SWT yang telah melimpahkan rahmat dan karunia-Nya, sehingga dapat diselesaikannya Petunjuk Praktikum ini. Sholawat dan salam semoga terlimpah kepada baginda Nabi Muhammad saw beserta keluarga, shohabat dan para pengikutnya.

Petunjuk Praktikum Pengenalan Pola ini disusun dengan harapan dapat memberikan kemudahan bagi mahasiswa untuk memahami dasar-dasar dalam membangun sebuah sistem pengenalan pola. Berbagai contoh aplikasi diberikan yang meliputi metode ekstraksi ciri maupun teknik-teknik klasifikasi. Ucapkan terima kasih yang sebesar-besarnya diberikan kepada pihak semua pihak yang telah membantu dalam proses penyusunan petunjuk praktikum ini.

Akhirnya, tiada gading yang tak retak, kritik dan saran membangun kami harapkan untuk lebih sempurnanya petunjuk praktikum ini. Semoga ilmu yang diperoleh dari membaca dan mempraktekkan petunjuk praktikum ini dapat bermanfaat dan barokah.

Yogyakarta, Februari 2019

Wassalam

Penulis

DAFTAR ISI

PENGANTAR MATLAB

Pertemuan ke : I
Alokasi Waktu : 1,5 Jam
Kompetensi Dasar : Mahasiswa mampu menggunakan software MATLAB

Indikator : 1. Membuat program sederhana menggunakan MATLAB dengan *.m file
2. Memahami statemen operasi kalang dan mampu membuat grafik.

A. Teori Pendukung

MATLAB (MATrix LABoratory) merupakan paket *software* yang dirancang untuk komputasi numeris dengan matriks. Pada MATLAB setiap angka dapat dinyatakan sebagai matriks sehingga lebih efisien dan mudah dibandingkan *software* lain seperti: Pascal, Fortran, Delphi, dll.

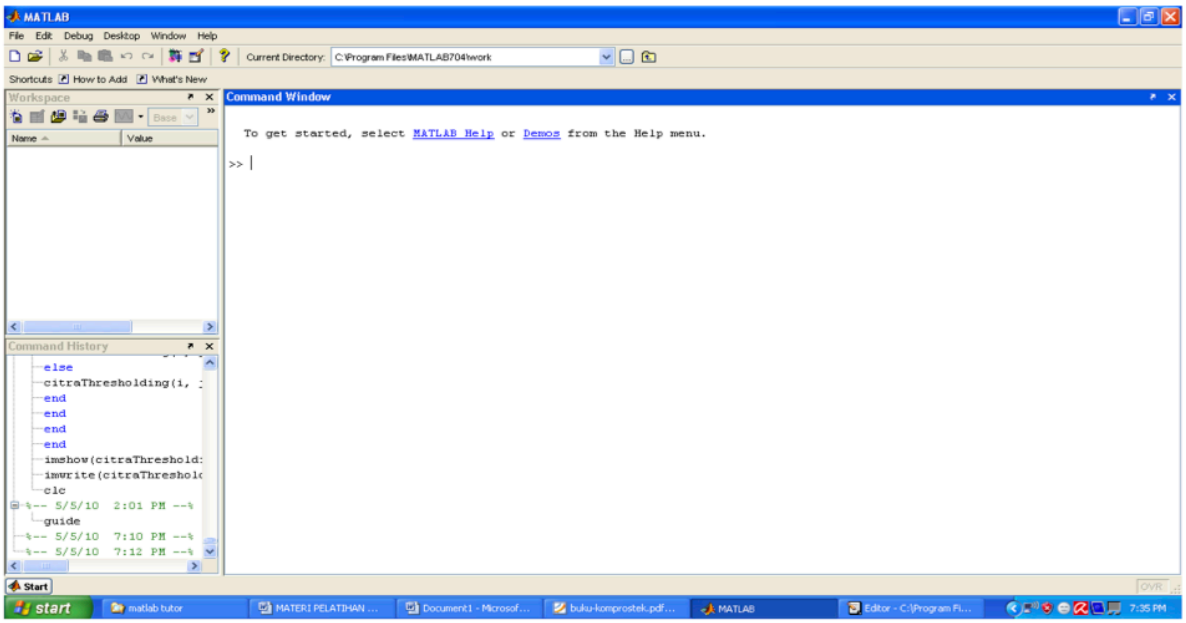
Beberapa keunggulan yang dimiliki MATLAB adalah:

1. Menyediakan perintah-perintah praktis untuk menyelesaikan persoalan-persoalan matematis seperti: operasi aljabar, operasi matriks, persamaan linear dan non-linear, persamaan diferensial, korelasi data, interpolasi data, dll.
2. Memungkinkan untuk membuat fungsi-fungsi sendiri disamping beberapa fungsi yang telah disediakan (*built in*).
3. Mampu menampilkan data dalam bentuk grafik 2D ataupun 3D
4. Memiliki toolbox untuk berbagai bidang seperti: *statistics, signal processing, image processing, neural network, fuzzy logic, etc.*
5. Memiliki menu demo dan tutorial yang memungkin pengguna dapat belajar sendiri dengan perintah: *help*

Memulai Program Matlab

Pada icon MATLAB double-click dan selanjutnya akan tampil Gambar 1. Secara umum lingkungan yang terdiri 3 bagian yaitu:

1. **Command Window**, merupakan jendela MATLAB tempat untuk mengeksekusi perintah, menampilkan masukan dan hasil.
   Dari prompt >> pernyataan atau statemen dituliskan dan untuk mengeksekusi perintah diakhir dengan menekan <ENTER>
2. **Workspace**, berfungsi menyimpan secara otomatis segala variabel masukan dan hasil
3. **Command history**, berfungsi menyimpan secara otomatis segala perintah yang telah ditulis pada command window



Gambar 1. Lingkungan MATLAB

Vektor dan Matriks

Vektor adalah merupakan matriks yang hanya terdiri dari satu baris atau satu kolom saja. Cara menuliskan matriks pada MATLAB dimulai dengan tutup buka ([) dan menggunakan koma (,) atau spasi untuk memisahkan elemen-elemen satu baris, sedangkan titik koma (;) untuk memisahkan elemen-elemen satu kolom serta diakhiri dengan tutup kurung (]).

1. Menyatakan suatu matriks

Misal matriks A:

$$A = \begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{bmatrix}$$

Cara menuliskan:

» A = [1 2 3; 4 5 6; 7 8 9]
» A(2,1)
» B = [1:1.5:6; 2 3 4 5]
» D =[]; D=[D;5]; D=[D;6;7]
» E = zeros(4, 5)
» F = ones(3, 4)
» G = rand(3,2)

2. Penjumlahan dan pengurangan suatu matriks

» b1 = [1 2 3]
» b2 = [4 5 6]

» b1 + b2
» b1 - b2
Jika ukuran vektor tidak sesuai maka akan mendapat *error*
» b = [1; 2; 3];
» c = [4 5 6];
» b+c
Error using ==> + Matrix dimensions must agree.

3. Perkalian
» b1
» b2
» b1*b2

4. Transpose matriks
» b2'
» b1*b2'

Berikan komentar anda dari hasil yang diperoleh dengan menjalankan perintah diatas! *Ingat aturan:* Jika anda mempunyai 2 matriks dengan ukuran $r_1 \times c_1$ dan $r_2 \times c_2$, anda dapat mengalikan keduanya dengan syarat $c_1 = r_2$.

» A = [1 2; 3 4; 5 6]
» b = [1;2;3]
» A*b
» A'*b

Perkalian elemen-elemen matriks dengan notasi '.*'
» b1.*b2

Perkalian skalar
» 2*c

5. Pembagian
Elemen per elemen :
» b1./b2
Skalar :
» c/2

6. Operator titik dua (:)
» x = 0:0.1:5 Menghasilkan vektor baris 51 elemen

7. Mensubstitusi elemen matriks
» A = [1 2 3; 4 5 6; 7 8 9]
» A(3,3) = 0

8. Operator titik dua (:) dalam matriks

```
» A(2,:)
» A(:,2:3)
```

Menghapus baris atau kolom dengan vektor kosong []

```
» A(:,2)=[]
```

9. Ukuran matriks

```
» size(A)
» [m,n]=size(A)
```

10. Determinan matriks

```
» det(A)
```

11. Invers matriks

```
» inv(A)
```

12. Penulisan lanjutan dalam matriks

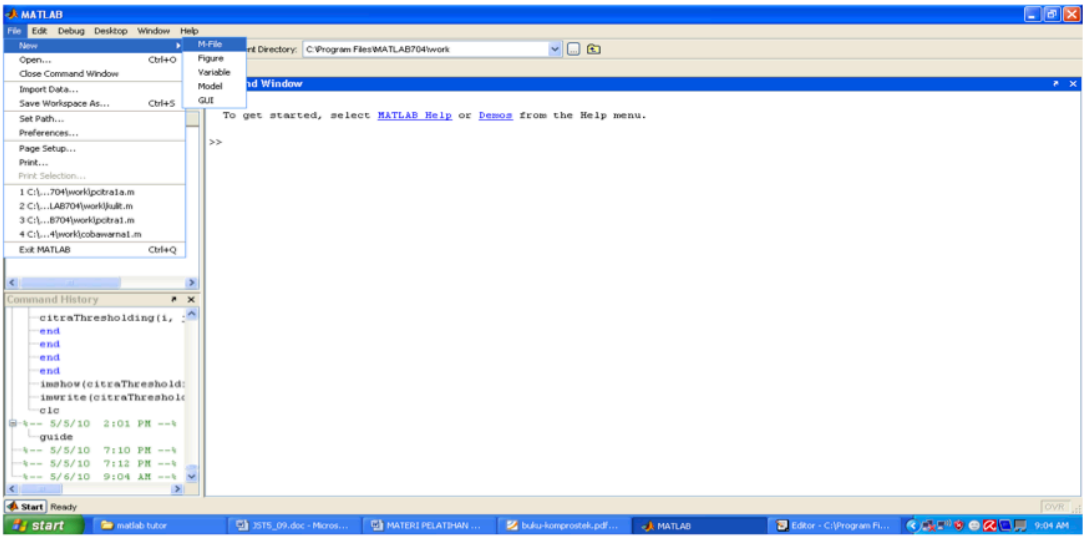
```
» G=[1 2 3 ...
      2 2 3 ...
      4 3 4]

» G=[1 2 3; ...
      2 2 3; ...
      4 3 4]
```

Perhatikan hasilnya!

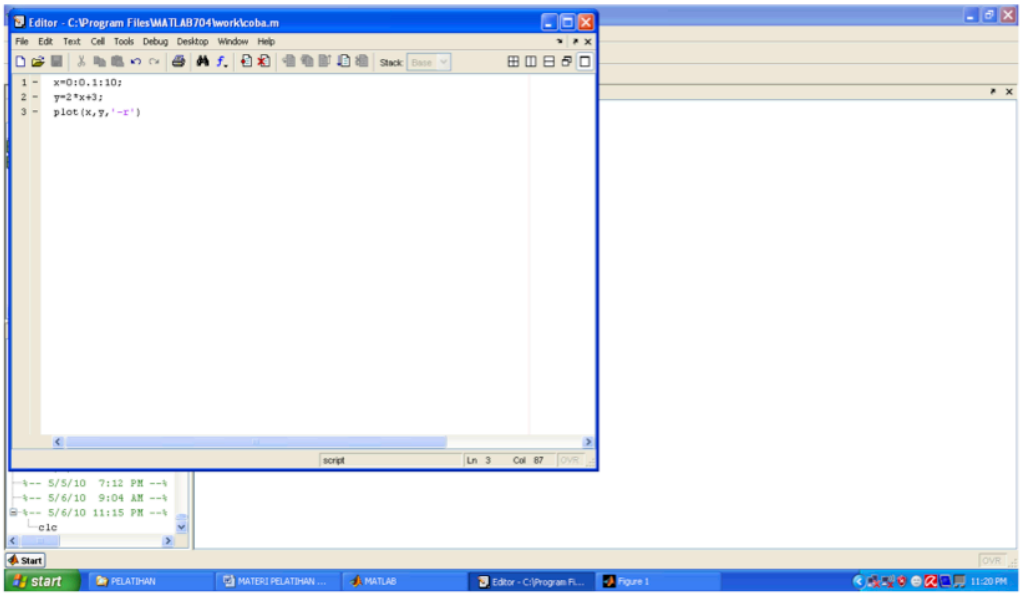
Membuat M-File

Pada saat ini dalam memasukkan perintah masih dilakukan melalui prompt >> command window. Untuk memudahkan dalam membuat program, pada MATLAB disediakan editor atau m-file untuk mengetikkan perintah-perintah dan menyimpannya. Penulisan program m-file dapat berupa skrip program atau fungsi.
Cara membuka layar editor mulai menekan klik pada File
→ New → M-File sebagaimana pada Gambar 2, berikut:



Gambar 2. Memulai membuka M-file

Setelah di-klik pada M-File akan tampir layar editor yang siap untuk menuliskan skrip program, sebagaimana pada Gambar 3 berikut:



Gambar 3. Layar editor

Skrip M-file

Skrip adalah sekumpulan perintah atau penyataan dalam suatu file yang sanat sederhana karena tidak mempunyai *input* argumen maupun *output* argument. Contoh skrip sederhana yang disimpan dengan nama file: coba.m

```
%Nama File: coba.m
%===============
clear all
x=0:0.1:10;
y=2*x+3;
plot(x,y,'-r')
```

Komentar pada m-file dapat dilakukan dengan menambahkan % sebelum menuliskan pernyataan. Hal ini berarti penyataan atau perintah yang diawali % tidak di-eksekusi. Menjalankan atau eksekusi program dapat dilakukan dengan beberapa cara:

- Tekan F5 pada keyboard
- Klik debug → run
- Aktifkan command window dan ketik nama file yang akan dijalankan kemudian tekan ENTER

  >> coba

Operasi kalang dan struktur perulangan (loop)

1. for ... end

Bentuk umum sintak nya :

```
for  variabel = ekspresi
        statemen-statemen
 end
```

Contoh 1:

```
for i=0:2:20
        disp(i)
end
```

Catat dan pahami hasil eksekusi-nya!

Contoh 2:

```
for i=1:4
        for j=1:3
              C(i,j)=i*j;
        end
end
```

Catat dan pahami hasil eksekusi-nya!

Contoh 3:

```
n = 5;
A = eye(n);
for j=2:n
    for i=1:j-1
        A(i,j)=i/j;
        A(j,i)=i/j;
    end
end
```

Catat dan pahami hasil eksekusi-nya!

2. while ... end

Bentuk umum sintak nya:

```
while   ekspresi
        statemen-statement
end
```

Contoh 4:

```
x = 1
while x <= 10
        x = 3*x
end
```

Catat dan pahami hasil eksekusi-nya!

3. if ... end

Struktur dapat bervariasi:

- if ... end
- if ... else ... end
- if ... elseif ... else ... end
-

Bentuk umum sintak nya:

```
if   ekspresi
        statemen-statement
end
```

Beberapa contoh didasarkan pada rumus kuadratis dengan variasi nilai diskriminan

```
1. D = b*b - 4*a*c;
        if D < 0
            disp('Peringatan: nilai diskriminan negatif, akar-akarnya imajiner');
        end
2. D = b*b - 4*a*c;
        if D < 0
            disp(' Peringatan: nilai diskriminan negatif, akar-akarnya imajiner');
        else
            disp('Akar-akar real, namun mungkin berbeda')
        end

3.  D = b*b - 4*a*c;
        if D < 0
            disp(' Peringatan: nilai diskriminan negatif, akar-akarnya imajiner');
        elseif D == 0
            disp('Nilai diskriminan nol, akar-akar
sama')
            else
                disp('akar-akar riil')
        end
```

B. Langkah Praktikum

1. Cotoh kasus 1:

Misalkan dua titik p1 dan p2 mempuyai koordinat masing-masing:

p1 = (1,5) p2=(4,7)

kita ingin menghitung jarak antara dua tersebut. Dengan menggunakan teorema Pythagoras, maka jarak d dapat dihitung dengan rumus:

$$d = \sqrt{s1^2 + s2^2}$$

$$= \sqrt{(4-1)^2 + (7-5)^2} = \sqrt{13} = 3{,}61$$

Penyelesaian dengan MATLAB dapat dibuat skrip program dengan nama file: kasus1.m

```
%Program menghitung jarak 2 titik
%Nama file : kasus1.m
%================================
                                ==
clc
clear
p1=[1,5];
p2=[4,7];
d=sqrt(sum((p2-p1).^2))
```

Jalankan program **kasus1.m** dan catat hasilnya!

2. Contoh kasus 2:

Sekumpulan data statistik berikut : [2.5 8.2 -1.1 -0.2 1.5]
Hitunglah (menggunakan kalkulator) :

- rerata (μ) $\mu = \frac{\sum_{i=1}^{N} x_i}{N}$
- variansi (σ^2) $\sigma^2 = \frac{\sum_{i=1}^{N} (x_i - \mu)^2}{N-1}$
- standar deviasi (σ) $\sigma = \sqrt{\sigma^2}$

Bandingkan hasilnya dengan cara membuat progran MATLAB dengan menggunakan fungsi-fungsi bawaannya, nama file: kasus2.m.

```
%Program Statistik
%Nama File : kasus2.m
%=================
clc
clear
x=[2.5 8.2 -1.1 -0.2 1.5];
disp('Data statistik ');
disp('rerata: ')
disp(mean(x))
disp('variansi: ')
disp(std(x)^2)
disp('standar deviasi: ')
disp(std(x))
```

Jalankan program **kasus2.m** dan catat hasilnya!

3. Fungsi M-file

Fungsi adalah m-file yang memiliki *input* argumen dan *output* argumen. Beberapa bagian dari sebuah fungsi untuk menghitung volume balok adalah

```
function hasil=volume(p,l,t)
```

dimana function adalah keyword hasil adalah *output* argumen
volume adalah nama fungsi
p,l,t adalah *input* argumen

Nama fungsi secara default sama dengan nama variabelnya. Pada Matlab juga telah ada fungsi-fungsi yang *built in* seperti: sin(x), cos(x), exp(x), log(x), mean(x), sum(x), std(x), dll.

Contoh kasus 3:
Menghitung volume bola dengan jari-jari sebagai *input* argumen. Rumus volume bola adalah:

$$V = \frac{4}{3}\pi . r^3$$

Maka untuk membuat fungsi nya adalah

```
function V=volumebola(r)
V=4/3*pi*r.^3
```

Bila fungsi diatas dijalankan dari command window, maka :

```
>> volumebola(5)

V =
    523.5988
```

Argumen fungsi dapat memiliki *output* argumen lebih dari satu dan nilainya ditulis dalam sebuah kurung siku. Sedangkan untuk *input* argumen ditulis dalam kurung biasa. Pemisahan antar argumen digunakan tanda koma.

Contoh kasus 4:
Mencari akar-akar dari persamaan kuadrat, didapat rumus:

$$x_{1,2} = \frac{-b \pm \sqrt{b^2 - 4ac}}{2a}$$

Fungsi akar yang akan dibuat mempunyai tiga argumen *input* yaitu a,b,c dan 2 argumen *output* yaitu x1 dan x2.

Maka untuk membuat fungsi nya adalah

```
function [x1, x2]=akar2(a,b,c)
x1=(-b+sqrt(b^2-4*a*c))/2*a;
x2=(-b-sqrt(b^2-4*a*c))/2*a;
```

Bila fungsi diatas dijalankan dari command window, maka :

```
>> [x1,x2]=akar2(1,-1,-12)
```

```
x1 =
     4
x2 =
    -3
```

4. Menampilkan satu grafik dengan perintah plot
   Contoh kasus 5:
   Sebuah fugsi sinus y=sin(x) dalam interval 0 ≤ x ≤ 2π. Buatlah m-file dengan nama kasus5.m, yang terdiri dari perintah-perintah berikut:

```
%Menampilkan satu grafik
%Nama File : kasus5.m
%=================
x= 0:pi/100:2*pi;
y= sin(x);
plot(x,y);
grid on
```

5. Menampilkan banyak grafik
   Contoh kasus 6:
   Plot grafik fungsi cosinus, y1 = 2 cos(x), y2 = cos(x), and y3 = 0,5 cos(x), dalam interval 0 ≤ x ≤ 2π. Buatlah m-file dengan nama kasus6.m, yang terdiri dari perintah-perintah berikut:

```
%Menampilkan Banyak Grafik
%Nama File : kasus6.m
%====================
x= 0:pi/100:2*pi;
y1 = 2*cos(x);
y2 = cos(x);
y3 = 0.5*cos(x);
plot(x,y1,'--',x,y2,'-',x,y3,':')
xlabel('0 \leq x \leq 2\pi')
ylabel('Nilai fungsi y(x)')
legend('2*cos(x)','cos(x)','0.5*cos(x)')
title('Grafik Fungsi-fungsi Cosinus')
axis([0 2*pi -3 3])
```

6. Menampilkan banyak grafik dalam satu gambar
   Contoh kasus 7:
   Buatlah m-file dengan nama kasus7.m, yang terdiri dari perintah-perintah berikut:

```
%Menampilkan Banyak Grafik dalam Satu Gambar
```

```
%Nama File : kasus7.m
%=====================================
x= 0: 0.01:2*pi;
y1 = sin(2*x);
y2 = cos(2*x);
y3 = 0.5*cos(5*x);
y4 = 2*sin(3*x);
subplot(2,2,1), plot(x,y1)
subplot(2,2,2), plot(x,y2)
subplot(2,2,3), plot(x,y3)
subplot(2,2,4), plot(x,y4)
```

C. Evaluasi

Nilai	Yogyakarta, ... Paraf asisten <.....

PEMROSESAN AWAL

Pertemuan ke : II
Alokasi Waktu : 1,5 Jam
Kompetensi Dasar : 1. Mahasiswa mampu pemroses awal pengenalan pola

Indikator : 1. Memahami cara pembacaan citra
2. Memahami pemrosesan awal
3. Memahami histogram sebagai ciri

A. Teori Pendukung

Pengenalan merupakan suatu hal yang mudah dilakukan oleh manusia, namun tidak demikian bagi sebuah mesin atau komputer. Pengenalan pola citra telah berkembang pesat seiring dengan kemajuan perangkat perangkat keras dan lunak komputer.

Beberapa contoh pengenalan citra seperti: tulisan tangan dan tandatangan, citra-citra medis, citra wajah, dll. Pada sistem pengenalan citra diawali dengan proses pengolahan citra kemudian dilanjutkan dengan ekstraksi citra dan klasifikasi pola citra

Pengolahan citra merupakan tahap awal atau pemrosesan awal dari suatu sistem pengenalan. Beberapa proses pengolahan citra yang penting dilakukan untuk pengenalan pola seperti: perubahan dari citra berwarna menjadi citra keabuan, konversi dari citra keabuan ke citra biner dengan metode pengambangan (*thresholding*). Sedangkan untuk ekstraksi ciri dapat digunakan metode histogram atau statistik dasar yang lain.

B. Langkah Praktikum

1. Pembacaan citra

   Pada matlab fungsi untuk melakukan pembacaan citra standar yaitu:

   imread('namafile')

   Setelah pembacaan citra dapat dilakukan proses seperti: mengubah citra berwarna RGB menjadi berderajat keabuan (*grayscale*), mengubah menjadi citra negatif dengan perintah dalam skrip berikut:

```
%Nama File: citra1.m
%====================
clear all;
close all;

I=imread('peppers.png');
figure (1)
imshow(I);
title('Citra asli');
```

```
figure (2)
G = rgb2gray(I);
imshow(G);
title('Citra keabuan');
```

Catat dan pahami fungsi bawaan Matlab **rgb2gray**
dengan eksekusi perintah
>> help rgb2gray

2. Membuat Konversi Citra berwarna menjadi Citra skala keabuan (*Grayscale*)
   Operasi konversi citra true color ke keabuan (*grayscale*) dengan rumus sebagai berikut:

$$W_{grayscale} = \frac{(R+G+B)}{3}$$

Contoh:

```
%Nama File: citra2.m
%=====================
clc
clg
figure (1)
Image= imread('peppers.png');
imshow(Image)
red=Image(:,:,1);
green=Image(:,:,2);
blue=Image(:,:,3);
gray=(red+green+blue)/3;
figure (2)
imshow(gray)
```

Perhatikan hasil eksekusinya dan bandingkan dengan hasil melalui penggunaan fungsi **rgb2gray**

3. Membuat Citra biner dari Citra skala keabuan (*Grayscale*)
   Suatu metode yang digunakan untuk mengubah citra denga format keabuan menjadi citra biner adalah pengambangan (*thresholding).* Rumus yang digunakan untuk konversi adalah

$$Ko = \begin{cases} 0, & jika\ Ki < ambang \\ 1, & jika\ Ki \geq ambang \end{cases}$$

atau

$$Ko = \begin{cases} 0, & jika\ Ki \geq ambang \\ 1, & jika\ Ki < ambang \end{cases}$$

Contoh:

```
%Nama File: ambang.m
%=====================
clear all
clc
clg
figure (1)
citra = imread('rice.png');
imshow(citra);
[baris, kolom] = size(citra);
citra = double(citra);
for i = 1:baris
   for j = 1:kolom
      if citra(i, j) < 120
            citraThresholding(i, j) = 0;
         else
            citraThresholding(i, j) = 1;
      end
   end
end
figure (2)
imshow(citraThresholding);
```

Ulangi meng-eksekusi file ambang.m diatas dengan mengganti nilai ambang menjadi 100 perhatikan hasilnya, dan ulangi untuk nilai ambang yang lain: 150 Jelaskan perbandingan atas hasil ketiganya!

4. Cropping dan Menyimpan Citra

Perintah untuk melakukan cropping (pemotongan bagian tertentu dari citra), menggunakan fungsi,

Imcrop(matriks citra, titiksudut_crop);

Perintah untuk melakukan penyimpanan citra menggunakan fungsi,

Imwrite(matriks citra,namafile,format ektensi);

Contoh:

```
%Nama File: citracrop.m
%=====================
clear all
I = imread('cameraman.tif');
I2 = imcrop(I,[60 40 100 90]);
```

```
figure, imshow(I), figure, imshow(I2)
        imwrite(I2,'camcrop.tif','tif')
```

Catat dan pahami hasil eksekusi-nya!

5. Membuat Histogram Citra

Perintah yang disediakan MATLAB untuk membuat histogram dari citra yaitu dengan fungsi,

imhist(matriks citra)

Contoh:

```
%Nama File: citrahist.m
%======================
clear all
I = imread ('rice.png');
GM = im2double(I);

%penambahan derau
final_length = length(GM);
noise = 0.25*randn(256);
im_noise=GM+noise;

%histogram
figure;
subplot(2,2,1);
imhist(GM);
title('histogram');

subplot(2,2,2);
imshow(GM);
title('Citra asli');

subplot(2,2,4);
imshow(im_noise);
title('Citra + derau');

subplot(2,2,3);
imhist(im_noise);
title('histogram Citra + derau');
```

6. Membuat citra bercampur derau

```
%Nama File: citranoise.m
%======================
```

```
clear all
I = imread ('cameraman.tif');
GM = im2double(I);

%penambahan derau
final_length = length(GM);
noise = 0.25*randn(256);
im_noise=GM+noise;
imwrite(im_noise,'camnoise.tif','tif')
```

C. Evaluasi

1. Buatlah fungsi *warna2abu.m* dengan cara memodifikasi file *citra2.m*, gunakan argumen input: Image dan argumen output: gray
2. Buatlah fungsi *abu2bin.m* dengan memodifikasi file *ambang.m* dengan argumen input: citra, nilai_ambang dan argumen output: citraThresholding
3. Buatlah file citra baru yang merupakan campuran citra asli **rice** dan noise yang berbeda-beda dengan cara memodifikasi file *citranoise.m*. Gunakan faktor pengali pada: noise = k*randn(256); dengan mengganti-ganti nilai k = 0.05, 0.1, 0.15, dan 0.2 selanjutnya simpan file citra baru dengan nama yang berbeda-beda pula.

Nilai	Yogyakarta, ... Paraf asisten <.....>
Jawaban Postest	

EKSTRAKSI CIRI

Pertemuan ke : III
Alokasi Waktu : 1,5 Jam
Kompetensi Dasar : 1. Mahasiswa mampu memproses ekstaksi ciri

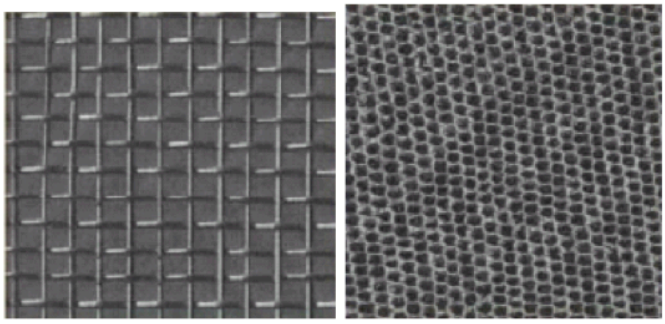
Indikator : 1. Memahami tekstur citra
2. Memahami proses ekstraksi ciri
3. Membuat pola citra

A. Teori Pendukung

Tekstur merupakan karakteristik intrinsik dari suatu citra yang terkait dengan tingkat kekasaran (roughness), granularitas (granulation), dan keteraturan (regularity) susunan struktural piksel. Aspek tekstural dari sebuah citra dapat dimanfaatkan sebagai dasar dari segmentasi, klasifikasi, maupun interpretasi citra.

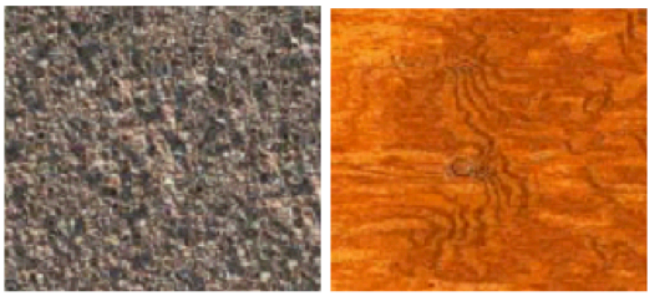
Tekstur dapat didefinisikan sebagai fungsi dari variasi spasial intensitas piksel (nilai keabuan) dalam citra. Berdasarkan strukturnya, tekstur dapat diklasifikasikan dalam dua golongan : makrostruktur dan mikrostruktur

Tekstur makrostruktur memiliki perulangan pola lokal secara periodik pada suatu daerah citra, biasanya terdapat pada pola-pola buatan manusia dan cenderung mudah untuk direpresentasikan secara matematis. Contoh tekstur makrostruktur sebagaimana pada Gambar 4, berikut:



Gambar 4. Citra tekstur makrostruktur

Pada tekstur mikrostruktur, pola-pola lokal dan perulangan tidak terjadi begitu jelas, sehingga tidak mudah untuk memberikan definisi tekstur yang komprehensif. Gambar 5 berikut ini menunjukkan contoh tekstur mikrostruktur.



Gambar 5. Citra tekstur mikrostruktur

Ekstraksi ciri orde pertama

Ekstraksi ciri orde pertama merupakan metode pengambilan ciri yang didasarkan pada karakteristik histogram citra. Histogram menunjukkan probabilitas kemunculan nilai derajat keabuan piksel pada suatu citra. Dari nilai-nilai pada histogram yang dihasilkan, dapat dihitung beberapa parameter ciri orde pertama, antara lain adalah mean, skewness, variance, kurtosis, dan entropy.

a. Mean (μ)
   Menunjukkan ukuran dispersi dari suatu citra

$$\mu = \sum_n f_n \, p(f_n)$$

   dimana f_n merupakan suatu nilai intensitas keabuan, sementara $p(f_n)$ menunjukkan nilai
   histogramnya (probabilitas kemunculan intensitas tersebut pada citra).

b. Variance (σ2)
   Menunjukkan variasi elemen pada histogram dari suatu citra

$$\sigma^2 = \sum_n (f_n - \mu)^2 \, p(f_n)$$

c. Skewness (α3)
   Menunjukkan tingkat kemencengan relatif kurva histogram dari suatu citra

$$\alpha_3 = \frac{1}{\sigma^3} \sum_n (f_n - \mu)^3 \, p(f_n)$$

d. Kurtosis (α4)
   Menunjukkan tingkat keruncingan relatif kurva histogram dari suatu citra

$$\alpha_4 = \frac{1}{\sigma^4} \sum_n (f_n - \mu)^4 \, p(f_n) \; - 3$$

e. Entropy (H)
   Menunjukkan ukuran ketidakaturan bentuk dari suatu citra

$$H = -\sum_{n} p(f_n) \cdot {}^{2}\log p(f_n)$$

B. Langkah Praktikum

1. Buatlah fungsi *forde1.m* yang dipergunakan untuk menghitung ciri orde satu dari sebuah citra:

```
function [m,v,s,k,e]=forde1(image)
H=imhist(image)';
H=H/sum(H);
I=[0:255];

m =  I*H';
v  = (I-m).^2*H';
s = (I-m).^3*H'/v^1.5;
k = (I-m).^4*H'/v^2-3;
e  = -H*log2(H+eps)';
```

2. Sediakan database citra tekstur makro struktur dan mikro struktur masing-masing 15 citra.
3. Ubahlah citra berwarna dari database menjadi format derajat keabuan dengan menggunakan fungsi **rgb2gray**
4. Tentukan ciri citra dengan menggunakan fungsi **forde1 .** Catat hasilnya dan berikan penjelasan dari hasil yang diperoleh.
5. Dengan melakukan seleksi ciri yaitu mean dan variance, gambarkan grafik mean Vs variance
6. Jelaskan apakah dua ciri yang dipilih telah mampu membedakan antara makro struktur dan mikrostruktur.

C. Evaluasi

Ulangi point 3-6 dengan menggunakan fungsi *warna2abu.m* yang telah dibuat pada praktikum 2.

Nilai	Yogyakarta, ... Paraf asisten <.....>

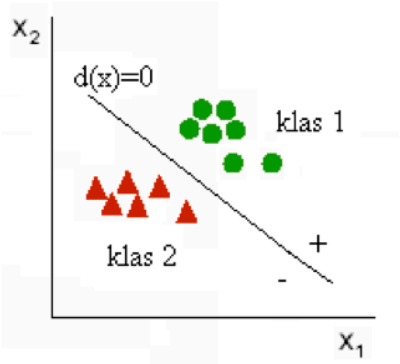
KLASIFIKASI DENGAN GARIS LINIER

Pertemuan ke : IV
Alokasi Waktu : 1,5 Jam
Kompetensi Dasar : Mahasiswa mampu memahami estimasi linier dan teknik klasifikasi sederhana

Indikator : 1. Memahami estimasi linier
2. Memahami pengklasifikasi garis linear

A. Teori Pendukung

Fungsi utama dari suatu sistem pengenalan pola adalah mengambil keputusan untuk menentukan anggota klas dari suatu pola-pola masukan. Olehkarenanya untuk melakukan tugas ini diperlukan beberapa aturan yang berdasarkan pada fungsi keputusan. Satu contoh sederhana fungsi keputusan untuk memisahkan populasi dua pola dapat digunakan persamaan gars linear, sebagaimana ditunjukkan pada Gambar 6 dibawah,



Gambar 6 Fungsi keputusan untuk mengklasifikasi dua jenis pola

Persamaan garis linear, misalnya $d(x) = w_1x_1 + w_2x_2 + w_3 = 0$, dimana w adalah parameter-parameter dan x_1, x_2 adalah variabel-variabel koordinat. Dari gambar 6 diatas, jelas bahwa pola-pola P = (x_1, x_2) pada klas 1 jika nilai-nilai variabel x nya disubstitusikan kedalam persamaan d(x) maka hasil bernilai positif. Dengan cara yang sama d(x) akan bernilai negatif jika pola-pola P = (x_1, x_2) pada klas 2 disubstitusikan kedalam persamaan d(x) tersebut. Olehkarena itu d(x) dapat digunakan sebagai fungsi keputusan, jika misalnya diberikan suatu pola **x** yang belum diketahui kelompok klas nya, maka dapat ditentukan bahwa pola **x** akan masuk dalam kategori klas 1 jika d(x) > 0, atau masuk kedalam kategori klas 2 jika d(x) <0.

B. Langkah Praktikum

1. Jalankan program *tekstur.m* dibawah ini:

```
%=====tekstur.m===
clc
clg
clear all
for i = 1:15
    citra1{i} = imread(['D: \datacitra\mikros\' num2str(i) '.jpg']);
```

```
    citra1{i}= rgb2gray(citra1{i});
    citra2{i} = imread(['D: \datacitra\makros\' num2str(i) '.jpg']);
    citra2{i}= rgb2gray(citra2{i});
end

rerata1=[ ];
var1=[ ];

rerata2=[ ];
var2=[ ];

for i = 1:15
    [m,v,s,k,e]=forde1(citra1{i});
    rerata1(i)=m;
    var1(i)=v;
    [rerata1]=[rerata1 rerata1(i)];
    [var1]=[var1 var1(i)];

    [m,v,s,k,e]=forde1(citra2{i});
    rerata2(i)=m;
    var2(i)=v;
    [rerata2]=[rerata2 rerata2(i)];
    [var2]=[var2 var2(i)];
end

plot(var1,rerata1,'*b')
hold on
plot(var2,rerata2,'or')
xlabel('variansi')
ylabel('rerata')
title('plot ekstraksi ciri')
hold off
```

Catat hasilnya dan tentukan persamaan garis untuk memisahkan antara klas citra makro struktur dan citra mikro struktur berdasarkan ciri rerata (mean) dan variansi.

2. Modifikasi program *tekstur.m* diatas dan tentukan persamaan garis untuk memisahkan antara klas citra makro struktur dan citra mikro struktur berdasarkan ciri rerata (mean) dan skewness.

C. Evaluasi

Ulangi untuk ciri rerata (mean) dan kurtosis serta ulangi lagi untuk ciri rerata (mean) dan entropy.

Nilai	Yogyakarta, ... Paraf asisten <.....>

KLASIFIKASI DENGAN METRIK JARAK

Pertemuan ke : V
Alokasi Waktu : 1,5 Jam
Kompetensi Dasar : Mahasiswa mampu memahami teknik klasifikasi menggunakan metrik jarak

Indikator : 1. Memahami proses klasifikasi pola
2. Memahami pengklasifikasi metrik jarak

A. Teori Pendukung

Satu ide dasar penggunaan fungsi jarak sebagai alat pengklasifikasi adalah kenyataan bahwa kemiripan atau perbedaan antara pola satu dengan dengan pola-pola yang lain dapat telah terkuantisasi dapat diukur nilai kemiripannya. Maka pengukuran kemiripan atau ketidakmiripan merupakan suatu dasar dalam tugas-tugas klasifikasi dan pengenalan. Salah satu ukuran kemiripan adalah dengan menentukan metrik jarak.

Ada beberapa metrik jarak yang cukup populer dan sering digunakan dalam sistem pengenalan pola yaitu: *Manhattan, Euclidean, Canberra,* dll.

Jika *x* dan *y* adalah dua vektor ciri d-dimensi (*x* = referensi/*template*, *y* = uji/*test*) maka metrik jarak dapat didefinisikan:

1. Metrik L1 (*Manhattan)*:

$$d_M(x,y) = \sum_{i=1}^{d} |x_i - y_i|$$

2. Metrik L2 (*Euclidean*):

$$d_E(x,y) = \sqrt{\sum_{i=1}^{d} (x_i - y_i)^2}$$

3. *Canberra:*

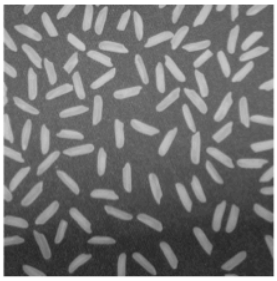
$$d_C(x,y) = \sum_{i=1}^{d} \frac{|x_i - y_i|}{|x_i| + |y_i|}$$

Contoh kasus:

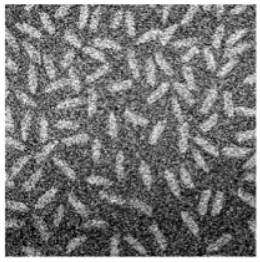
Suatu sistem pengenalan akan dilakukan klasifikasi dengan mengukur kemiripan suatu pola citra. Adapun citra referensi terdiri dari dua kelas yaitu *rice* dan *cameraman*, dan beberapa citra yang tercampur *noise,* untuk pengujian kemiripannya sebagai berikut:

a). Gb. Cameraman

b). Gb. Cameraman ber-noise



c). Gb. Rice



d) Gb. Rice ber-noise

B. Langkah Praktikum

1. Buatlah fungsi untuk menghitung jarak dengan metode Manhattan.

```
function d=dmanhattan(x,y)
    d=sum(abs(x-y));
end
```

2. Buatlah program **kenalcitra.m,** dibawah ini:

```
%Nama File: kenalcitra.m
%====================
clc;
clear;
I1=imread('rice.png');
I2=imread('cameraman.tif');
Im1=imresize(I1,[20 20],'bilinear');
x1=reshape(Im1,[400,1]);
Im2=imresize(I2,[20 20],'bilinear');
x2=reshape(Im2,[400,1]);
```

```
%test
I=imread('ricenoise.tif');
Im=imresize(I,[20 20],'bilinear');
y=reshape(Im,[400,1]);

d1=dmanhattan(x1,y)
d2=dmanhattan(x2,y)

minimum=min([d1 d2]);
if (d1==minimum)
    class='rice'
elseif (d2==minimum)
    class='cameraman'
end
```

Jalankan program **kenalcitra.m** diatas dan cacat serta perhatikan hasilnya!

2. Modifikasi program **kenalcitra.m** dengan menggunakan database datacitra mikro struktur dan makro struktur, sebagai citra referensi masing-masing data nomer 1.

3. Buatlah pengujian dengan menggunakan citra makro dan mikro struktur no.2 -11, untuk mendapatkan unjuk kerja sistem dengan mengisi tabel berikut:

Masukan (citra)	Fungsi Jarak Manhattan (L1)	
	Mikro struktur	Makro struktur
Mikro struktur		
Makro struktur		
Akurasi (%) :		

C. Evaluasi

Buatlah fungsi **distL2.m** untuk menghitung jarak dengan metode Euclidean. Buatlah pengujian dengan menggunakan citra makro dan mikro struktur no.2 -11, untuk mendapatkan unjuk kerja sistem dengan mengisi tabel diatas:

Nilai	Yogyakarta, ... Paraf asisten <.....>

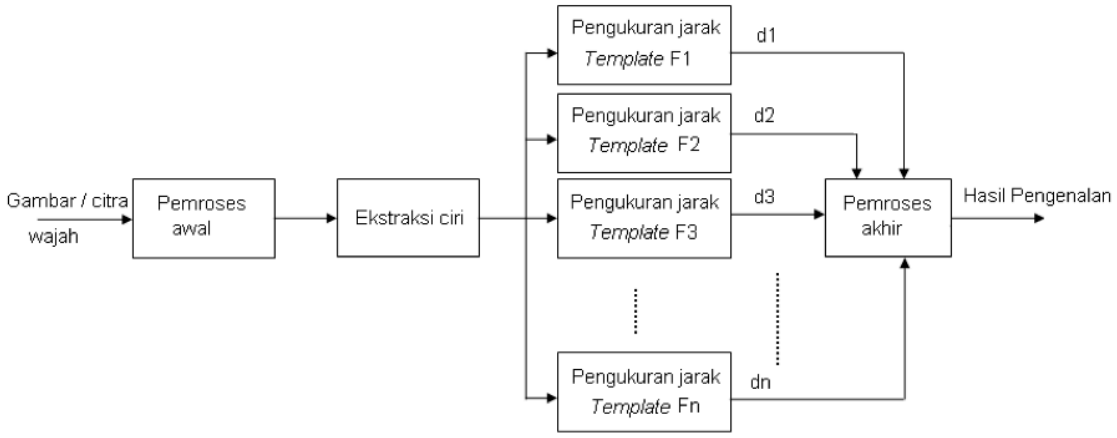
SISTEM PENGENALAN WAJAH

Pertemuan ke : VI
Alokasi Waktu : 1,5 Jam
Kompetensi Dasar : Mahasiswa mampu memahami sistem pengenalan wajah

Indikator : 1. Memahami proses pengenalan wajah sederhana
2. Memahami pengklasifikasi metrik jarak

A. Teori Pendukung

Wajah merupakan salah satu karakteristik biometrik yang digunakan untuk mengenali seseorang selain karakteristik yang lain seperti ucapan, sidik jari, retina, dll. Praktikum ini memfokuskan pada perbandingan efektifitas penggunaan fungsi jarak dengan sebagai pengklasifikasi pada sistem biometrik pengenalan wajah. Ketiga fungsi jarak yang dikaji yaitu: Manhattan (L1), Euclidean (L2) dan Canberra. Secara umum sistem pengenalan wajah dapat ditunjukkan pada Gambar 6, berikut:



Gambar 6. Blok diagram sistem pengenalan wajah

Pada prinsipnya sistem pengenalan wajah meliputi 5 bagian yaitu: akuisisi data, pemproses awal, ekstraksi ciri, pengklasifikasi dan pemroses akhir.

Akuisisi data dan *Database*

Proses awal dari pengenalan wajah adalah akuisisi data yaitu pengambilan gambar wajah menggunakan kamera digital atau *webcam*. Pada praktikum ini data yang digunakan diperoleh melalui download dari:
http://www.ee.surrey.ac.uk/Personal/T.Windeatt/msc_projects/. Data terbagi dalam 2 bagian yaitu data untuk pelatihan (*training set)* dan data untuk pengujian (*testing set*), dari 5 gambar wajah orang dengan notasi: F1, F2, ….dan F5 dalam berbagai ekspresi. Masing-masing wajah terdiri 8 sampel data. Data nomor 1-4 untuk pelatihan (*training*) sedangkan data nomor 5-8 untuk pengujian (*testing*).

Pemroses awal dan ekstraksi ciri

Pemroses awal pada dasarnya bertujuan untuk mendapatkan data dengan ukuran yang lebih kecil namun cukup mewakili data asli yang sebenarnya. Pada data gambar wajah yang digunakan dalam penelitian ini akan dilakukan penurunan resolusi gambar asli yaitu dari 112×92 piksel menjadi 30×20 piksel. Selanjutnya gambar wajah yan juga dapat dikatakan sebagai matriks berukuran 30×20 tersebut diubah menjadi suatu bentuk pola berupa matriks berukuran 600×1.

Ekstraksi ciri bertujuan untuk menajamkan perbedaan-perbedaan pola sehingga akan sangat memudahkan dalam pemisahan kategori klas pada proses klasifikasi. Gambar/citra wajah setelah melalui pemroses awal dapat dinyatakan dalam bentuk pola :

$$\mathrm{Im} = [\Gamma_1, \Gamma_2, ..., \Gamma_n]$$

Rerata pola citra Ψ dan pola dengan pengurangan reratanya Φ, didefinisikan dengan:

$$\psi = \frac{1}{M}\sum_{n=1}^{M}\Gamma_n$$

$$\Phi_i = \Gamma_i - \psi$$

Φ_i merupakan pola hasil ekstraksi ciri yang akan digunakan sebagai masukan pada proses pengklasifikasi.

Pengklasifikasi dan Pemroses Akhir

Satu ide dasar penggunaan fungsi jarak sebagai alat pengklasifikasi adalah kenyataan bahwa kemiripan atau perbedaan antara pola satu dengan pola yang lain dapat diukur nilai kemiripannya. Maka pengukuran kemiripan atau ketidakmiripan merupakan suatu dasar dalam tugas-tugas klasifikasi dan pengenalan. Pada sistem pengenalan wajah yang dibuat ini akan dibandingkan efektifitas dua fungsi jarak yaitu Manhattan (L1), Euclidean (L2) dan Canberra, sebagai pengklasifikasi. Ketiga fungsi jarak tersebut sebagaimana terdapat pada praktikum sebelumnya.

Pola template merupakan pola yang akan digunakan sebagai referensi pada proses pengukuran jarak. Pola ini didasarkan pada data pelatihan (*training set*) yang digunakan dengan menghitung rerata dari sejumlah N data pelatihan. Pola template wajah untuk masing-masing orang dapat diperoleh dari:

$$\overline{\Phi}_i^k = \frac{1}{N}\sum_{k=1}^{N}\Phi_i^k$$

Pemroses akhir merupakan suatu proses tahap terakhir untuk pengambilan keputusan hasil pengukuran jarak atau tingkat kemiripan. Suatu pola baru yang belum dikenal oleh sistem dapat dikatakan mirip dengan salah satu pola template/referensi jika telah dilakukan proses penghitungan nilai jarak antara pola baru tersebut dengan setiap pola template/referensi. Kategori kemiripan didasarkan pada nilai jarak minimum, yang dapat didefinisikan:

$$k^* = \arg \min_k d_k, \quad 1 \le k \le n$$

dimana, n adalah jumlah orang yang akan dikenali.

B. Langkah Praktikum

1. Buatlah program KenalWajah.m

```
%============================================================
%Nama file : KenalWajah.m
%============================================================
% Membaca data wajah untuk pelatihan dari 5 orang yang berbeda
% dengan masing-masing menggunakan 4 sampel data
clc;
Clear;
for i = 1:4
    data1{i} = imread(['C:\MATLAB7\work\face\f1\' num2str(i) '.tif']);
    data2{i} = imread(['C:\MATLAB7\work\face\f2\' num2str(i) '.tif']);
    data3{i} = imread(['C:\MATLAB7\work\face\f3\' num2str(i) '.tif']);
    data4{i} = imread(['C:\MATLAB7\work\face\f4\' num2str(i) '.tif']);
    data5{i} = imread(['C:\MATLAB7\work\face\f5\' num2str(i) '.tif']);
end;

% Pemrosesan awal dan ekstraksi ciri
for i = 1:4
    data1{i} = imresize(data1{i},[30 20]);
    x1{i}=reshape(data1{i},[600,1]);
    x1{i}=x1{i}-mean(x1{i});
    x1{i}=
    data2{i} = imresize(data2{i},[30 20]);
    x2{i}=reshape(data2{i},[600,1]);
    x2{i}=x2{i}-mean(x2{i});

    data3{i} = imresize(data3{i},[30 20]);
    x3{i}=reshape(data3{i},[600,1]);
    x3{i}=x3{i}-mean(x3{i});

    data4{i} = imresize(data4{i},[30 20]);
    x4{i}=reshape(data4{i},[600,1]);
    x4{i}=x4{i}-mean(x4{i});

    data5{i} = imresize(data5{i},[30 20]);
    x5{i}=reshape(data5{i},[600,1]);
    x5{i}=x5{i}-mean(x5{i});
end;

% Template 5 wajah ('training mode')
p1=mean([x1{1} x1{2} x1{3} x1{4}],2);
p2=mean([x2{1} x2{2} x2{3} x2{4}],2);
p3=mean([x3{1} x3{2} x3{3} x3{4}],2);
p4=mean([x4{1} x4{2} x4{3} x4{4}],2);
p5=mean([x5{1} x5{2} x5{3} x5{4}],2);

%'testing mode'
I=imread(['C:\MATLAB7\work\face\f1\5.tif']);
x= imresize(I,[30 20]);
x=reshape(x,[600,1]);
px=double(x-mean(x));
```

```
d1=distL1(px,p1);
d2=distL1(px,p2);
d3=distL1(px,p3);
d4=distL1(px,p4);
d5=distL1(px,p5);

%Pengambilan keputusan hasil pengenalan
minimum=min([d1 d2 d3 d4 d5]);
if (d1==minimum)
    class='f1'
    elseif (d2==minimum)
        class='f2'
        elseif (d3==minimum)
            class='f3'
            elseif (d4==minimum)
                class='f4'
                else (d5==minimum)
                class='f5'
end
```

3. Jalankan dan Catat hasil pengamatannya, ulangi dengan mengganti masukan file citra yang diujikan dari masing-masing orang (F1-F4) pada nomor citra 5-8. Sajikan hasilnya dengan mengisi tabel berikut:

Masukan	Fungsi Jarak Manhattan (L1)				
	F1	F2	F3	F4	F5
F1					
F2					
F3					
F4					
F5					
Akurasi (%)					

C. Evaluasi

1. Buatlah fungsi **distL2.m** untuk menghitung jarak dengan metode Euclidean. Kerjakan seperti pada point 3 dengan menggunakan metode Euclidean dan sajikan hasilnya dengan mengisi tabel serupa diatas.

2. Buatlah fungsi **canberra.m** untuk menghitung jarak dengan metode Canberra. Kerjakan seperti pada point 3 dengan menggunakan metode Canberra dan sajikan hasilnya dengan mengisi tabel serupa diatas.

Nilai	Yogyakarta, ... Paraf asisten <.....>

JARINGAN SYARAF TIRUAN UNTUK PENGENALAN POLA

Pertemuan ke : VII
Alokasi Waktu : 1,5 Jam
Kompetensi Dasar : Mahasiswa mampu memahami jaringan syaraf tiruan

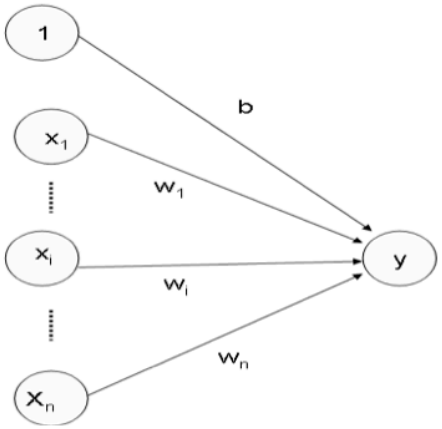
Indikator : 1. Memahami konsep jaringan syaraf tiruan
2. Memahami algoritma hebb
3. Memahami aplikasi algoritma hebb pengenalan pola

A. Teori Pendukung

Jaringan saraf tiruan (JST) merupakan sistem pemroses informasi yang unjuk kerjanya didasarkan pada jaringan syaraf biologis, yangtelah dikembangkan dengan beberapa asumsi-asumsi: proses informasi terjadi pada banyak elemen sederhana yang disebut neuron, sinyal antara neuron lewat link koneksi yang mana setiap link koneksi mempunyai bobot. Setiap neuron mempunyai fungsi aktifasi input yang merupakan jumlah bobot sinyal masukan untuk menentukan sinyal output.
Karakteristik JST terdiridari:

4. Model koneksi antar neuron (arsitektur)
5. Metode penentuan bobot koneksi (algoritma pelatihan)
6. Fungsi aktifasi

Arsitektur:



Gambar 2: Arsitektur JST sederhana

Algoritma Pembelajaran Hebb

Pelatihan dengan memodifikasi kekuatan sinapsis (bobot) yaitu bila 2 neuron yang terhubung dan keduanya pada kondisi 'hidup' pada saat yang sama, maka bobot

tersebut harus ditambah. Apabila data disajikan secara bipolar, maka perbaruan bobotnya adalah:

$$w_i(baru) = w_i(lama) + x_i y$$

Algoritma:

L0. Inisialisasi semua bobot:

$w_i = 0;$ dengan *i*= 1,2,...,n

L1. Untuk setiap pasangan vektor pembelajaran input-output lakukan langkah 2 - 4.

L2. Tetapkan aktivasi unit input:

$x_i = s_i;$ dengan *i*= 1,2,...,n

L3. Tetapkan aktivasi unit output:

$y = t;$

L4. Atur bobot:

$w_i(baru) = w_i(lama) + x_i y$ dengan *i*= 1,2,...,n

Atur bias:

$b(baru) = b(lama) + y$

Fungsi aktivasi

Beberapa fungsi aktivasi dirumuskan sebagai:

a. Fungsi Bipolar

$$f(x_j) = \begin{cases} 1 & x_j \geq 0 \\ -1 & x_j < 0 \end{cases}$$

b. Fungsi Saturating Linear

$$f(x_j) = \begin{cases} 0 & x_j \leq 0 \\ x_j & 0 < x_j < 1 \\ 1 & x_x \geq 1 \end{cases}$$

c. Fungsi Sigmoid

$$f(x_j) = \frac{1}{1 + e^{-\lambda_j x_j}}$$

B. Langkah Praktikum

1. Memahami fungsi Aktivasi

a. Fungsi bipolar dengan threshold (θ)

$$y = f(x) = \begin{cases} 1; & jika \quad x > \theta \\ 0; & jika \quad -\theta \le x \le \theta \\ -1; & jika \quad x < -\theta \end{cases}$$

Buatlah fungsi bipolar2.m file dari fungsi diatas

```
function y=bipolar2(x,th)
    if x > th
        y=1;
    elseif x < -1*th
        y=-1;
    else
        y=0;
    end
end
```

Catatlah hasil nya dengan menggunakan fungsi *bipolar2.m* dengan threshold = 0.5 untuk:

i. f(0.8, 0.5) = …. iii. f(-0.2, 0.5) =…..
ii. f(35, 0.5) = ….. iv. f(-4.5, 0.5) = …

b. Fungsi Sigmoid

$$y = f(x) = \frac{1}{1 + e^{-\alpha x}}$$

Buatlah fungsi sigmoid.m file dari fungsi diatas

```
function y=sigmoid(x)
     a=0.5;
    y=1./(1+exp(-1*a.*x))
end
```

Catatlah hasil nya dengan menggunakan fungsi *sigmoid.m* untuk:

i. f(0.2) = …. iii. f(2) =…..
ii. f(35) = ….. iv. f(4.5) = …

c. Fungsi Saturating Linear

$$y = f(x) = \begin{cases} 1 & ; \quad x \ge 1 \\ x & ; \quad -1 \le x \le 1 \\ -1 & ; \quad x \le -1 \end{cases}$$

Buatlah fungsi *.m file dari fungsi diatas

Catatlah hasil nya dengan menggunakan fungsi *satlin.m* untuk:

i. f(0.2) = …. iii. f(-0.3) =…..
ii.f(3.5) = ….. iv. f(-5.5) = …

d. Fungsi Sigmoid bipolar

$$y = f(x) = \frac{1 - e^{-x}}{1 + e^{-x}}$$

Buatlah fungsi *.m file dari fungsi diatas

Catatlah hasil nya dengan menggunakan fungsi *sigbip.m* untuk:

i. f(0.2) = …. iii. f(-0.3) =…..
ii.f(0.3) = ….. iv. f(-0.5) = …

2. Buatlah fungsi lhebb.m

```
function [w,b]=lhebb(pm,pt)
%algoritma Hebb
%Input : pm = pola-pola masukan
%        pt = pola-pola target
%Output : bobot
% inisialisasi semua bobot nol
w= rand(1,length(pm(1,:)));
w= w-w
b=0;
for k=1:length(pm(:,1))
   disp('data ke'),k
   s= pm(k,:);
   x=s;
   t=pt(k,:);
   y=t';
   % Perbarui bobot
   w = w + (y * x)
   b = b + y
end
```

3. Buatlah fungsi simhebb.m

```
function y=simhebb(px,w)

yin = px*w';
if yin > 0
   y=1;
```

```
else
    y=-1;
end
```

C. Evaluasi

Jalankan program prak7.m berikut:

```
%Nama file: prakt7.m
%===============

%Mode training
clc
clear
pm=[-1 -1;-1 1; 1 -1;1 1];
pt=[-1; 1; 1; 1];
w=lhebb(pm,pt)

save bobot.mat w

%Mode testing
%---------------------
Px=input('Masukkan pola data yang diujikan: ')
load bobot.mat
w=bobot
y=simhebb(Px,w)
```

Catat hasilnya

Nilai	Yogyakarta, ... Paraf asisten <.....>

ALGORITMA PERCEPTRON UNTUK PENGENALAN POLA

Pertemuan ke : VIII
Alokasi Waktu : 1,5 Jam
Kompetensi Dasar : Mahasiswa mampu memahami algoritma perceptron

Indikator : Memahami konsep algoritma perceptron untuk pengenalan pola

A. Teori Pendukung

Perceptron biasanya digunakan untuk mengklasifikasikan pola tertentu dan juga dikenal sebagai pemisah secara linear. Perceptron mempunyai 3 lapis neuron yaitu: unit sensori, unit asosiator dan unit tanggapan.

Output perceptron: $y = f(y_{\_in})$

dengan fungsi aktivasi:

$$f(y_{\_in}) = \begin{cases} 1 & bila \quad y_{\_in} > \theta \\ 0 & bila \quad -\theta \le y_{\_in} \le \theta \\ -1 & bila \quad y_{\_in} < \theta \end{cases}$$

Implementasi dalam MATLAB

```
function y=bipolar2(x,th)
   if x > th
      y=1;
   elseif x < -1*th
      y=-1;
   else
      y=0;
   end
end
```

Perubahan bobot:

$w_i(baru) = w_i(lama) + \alpha.t.x_i$

dengan nilai target t= +1 atau -1 dan α = laju pelatihan

Algoritma Perceptron (MISO):

L0. Inisialisasi bobot dan bias (untuk mudahnya tetapkan semua bobot dan bias sama dengan nol)

Tetapkan laju pelatihan α (0<α≤1) dan (untuk mudahnya α = 1)

L1. Selama syarat berhenti bernilai *false*, kerjakan langkah 2-6,

L2. Untuk setiap pasangan pembelajaran s:t, kerjakan langkah 3-5,

L3. Tetapkan aktivasi unit input

$x_i = s_i;$ dengan *i*= 1,2,...,n

L4. Hitung tangggapan unit output

$$y\_in = b + \sum_{i=1}^{n} x_i w_i$$

$$y = \begin{cases} 1; & jika \quad y\_in > \theta \\ 0; & jika \quad -\theta \le y\_in \le \theta \\ -1; & jika \quad y\_in < -\theta \end{cases}$$

L5. Perbarui bobot dan bias jika terjadi *error*:

Bila y ≠ t maka:

$$w_i(baru) = w_i(lama) + \alpha.t.x_i$$

$$b(baru) = b(lama) + \alpha.t$$

Bila y = t, maka:

$$w_i(baru) = w_i(lama)$$

$$b(baru) = b(lama)$$

L6. Uji syarat berhenti; jika tidak terjadi perubahan bobot pada langkah 2, maka kondisi berhenti *true*, namun bila masih terjadi perubahan maka kondisi berhenti *false.*

```
function [w,b,err]=lperceptron1(pm,pt,lp,maxiter)
%Input : pm = pola-pola masukan
%        pt = pola-pola target
%        lp = laju pembelajaran
%        maxiter = jumlah maksimum iterasi
%        eps = toleransi error
%Output : w = bobot
%         b = bias
%         err = error
%=========================================
% inisialisasi semua bobot dan bias nol
w= rand(1,length(pm(1,:)));
w=w-w;
b=0;
n=0;
MSE=1;
err=[];
while (n < maxiter)
   disp('iterasi ke:'),n+1
   for k=1:length(pm(:,1))
      disp('data ke:'),k
      s= pm(k,:);
      x=s;
      yin=b+sum(x*w');
      y=bipolar2(yin,0.2);
      t=pt(k,:);
```

```
        if y ~= t
            % Perbarui bobot
            w= w + (lp*t* x)
            b= b+(lp*t)
        else
         % Perbarui bobot
           w=w
           b=b
        end

       e(k)=(y-t)
    end
    n=n+1;
   err(n)=sqrt(sum(e.^2))/(length(pm(:,1))*n);
   MSE=err(n)
end
```

Algoritma Perceptron MIMO (Multi Input Multi Output):

L0. Inisialisasi bobot dan bias (untuk mudahnya tetapkan semua bobot dan bias sama dengan nol)

Tetapkan laju pelatihan α (0<α≤1) dan (untuk mudahnya α = 1)

L1. Selama syarat berhenti bernilai *false*, kerjakan langkah 2-6, {jumlah iterasi / toleransi error}

L2. Untuk setiap pasangan pembelajaran s:t, kerjakan langkah 3-5,

L3. Tetapkan aktivasi unit input

$x_i = s_i;$ dengan *i*= 1,2,...,n

L4. Hitung tangggapan unit output

$$y\_in_j = b_j + \sum_{i=1}^{n} x_i w_{ij}$$

$$y_j = \begin{cases} 1; & jika \quad y\_in_j > \theta \\ 0; & jika \quad -\theta \le y\_in_j \le \theta \\ -1; & jika \quad y\_in_j < -\theta \end{cases}$$

L5. Perbarui bobot dan bias jika terjadi *error*:

Bila y ≠ t maka:

$$w_{ij}(baru) = w_{ij}(lama) + \alpha.t_j.x_i$$

$$b_j(baru) = b_j(lama) + \alpha.t_j$$

Bila y = t, maka:

$$w_{ij}(baru) = w_{ij}(lama)$$

$$b(baru) = b(lama)$$

L6. Uji syarat berhenti; jika tidak terjadi perubahan bobot pada langkah 2, maka kondisi berhenti *true*, namun bila masih terjadi perubahan maka kondisi berhenti *false.*

```
function [w,b,err]=lperceptron2(pm,pt,lp,maxiter)
%Input : pm = pola-pola masukan
%        pt = pola-pola target
%        lp = laju pembelajaran
%        maxiter = jumlah maksimum iterasi
%        eps = toleransi error
%Output : w = bobot
%         b = bias
%         err = error
%=========================================
% inisialisasi semua bobot dan bias nol
n=length(pm(1,:));
m=length(pt(1,:));
w= zeros(n,m);
b=zeros(1,m);
err=[];
while  (n < maxiter);
for iter=1:maxiter
    disp('iterasi ke:'),iter
    for k=1:length(pm(:,1))
        disp('data ke:'),k
for i=1:n;
                        s(i)= pm(k,i);
                        x(i)=s(i);
     end
for j=1:m;
yin(j) =b(j)+sum(x(i)*w(i,j)');
                        y(j)=bipolar2(yin(j),0.5);
     end
        if y(j) ~= t(j)
            % Perbarui bobot
            w(i,j)= w(i,j) + (lp*t(j)* x(i,j))
            b(j)= b(j)+(lp*t(j))
        else
         % Perbarui bobot
           w(i,j)=w(I,j)
           b(j)=b(j)
        end
    end
end
```

B. Langkah Praktikum

1. Lakukanlah pelatihan dan pengujian fungsi AND dengan laju pembelajaran α = 1 dan iterasi 3. Buat dan jalankan **Prak81a.m**

```
%Prak81a.m
%Data pelatihan
%--------------------
clear
pm=[1 1;1 0;0 1;0 0];
pt=[1;0;0;0];

%Mode training
% inisialisasi semua bobot nol
[w,b,err]=lperceptron1(pm,pt,1,3)
save bobotp.mat w b
```

Buat dan jalankan **Prak81b.m** dengan memberikan masukan masing-masing data yang digunakan pelatihan.

```
%Prak81b.m
%Mode testing
%----------------------
clear
load bobotp.mat
Px=input('Masukkan pola data yang diujikan: ')
yin=b+Px*w'
Th=1;  % pada percobaan diganti 2, 3,...
y=bipolar2(yin, Th)
```

Isilah tabel hasil percobaan berikut:

x1	x2	yin	Th =1	Th = 2	Th = 3
			y	y	y
1	1				
1	0				
0	1				
0	0				

Dari hasil percoaan diperoleh bahwa data yang telah dilatihkan akan dapat dikenali dengan baik bila dipilih nilai threshold .....

2. Dengan cara seperti pada percobaan 1 diatas, lakukan pelatihan dan pengujian untuk pasangan data pola masukan dan pola target dibawah ini.

pm (pola masukan)				pt (pola target)	
x1	x2	x3	x4	t1	t2
0	0	0	0	1	0
0	0	0	1	1	0
0	0	1	0	1	0
0	0	1	1	1	0
1	1	0	0	0	1
1	1	0	1	0	1
1	1	1	0	0	1
1	1	1	1	0	1

a. Buatlah program dengan nama file **Prak82a.m** dengan cara memodifikasi program **Prak81a.m** untuk proses pelatihan dan gunakan fungsi **lperceptron2.m** dan berikan iterasi = 10.
b. Buatlah program dengan nama file **Prak82b.m** dengan cara memodifikasi program **Prak81b.m** untuk proses pengujian (testing)
c. Isilah tabel hasil percobaan berikut:

x1	x2	x3	x4	yin1	Yin2
0	1	0	0		
0	0	0	1		
0	1	1	0		
0	0	1	1		
1	1	0	0		
1	0	0	1		
1	1	1	0		
1	0	1	1		

Tentukan nilai threshold yang dipilih agar data pelatihan dapat dikenali dengan baik.

C. Evaluasi

Nilai	Yogyakarta, ... Paraf asisten <.....>

ALGORITMA BACKPROPAGATION UNTUK PENGENALAN POLA

Pertemuan ke : IX
Alokasi Waktu : 1,5 Jam
Kompetensi Dasar : Mahasiswa mampu memahami algoritma backpropagation

Indikator : Memahami konsep algoritma backpropagation untuk pengenalan pola

A. Teori Pendukung

Backpropagation (BP) merupakan algoritma pembelajaran yang terdiri dari tiga bagian yaitu: umpan maju pola pelatihan, perhitungan perambatan balik dan pengaturan bobot. Algoritma ini menggunakan *error* output untuk mengubah nilai bobot-bobotnya dalam arah balik (*backward*). Nilai *error* diperoleh dari proses sebelumnya yaitu pada tahap perambatan maju (*forward*). Inti dari algoritma belajar propagasi balik ini terletak pada kemampuannya mengubah nilai-nilai bobotnya untuk merespon adanya *error.*

Untuk dapat menghitung error, sampel data yang digunakan sebagai pembelajaran harus mengandung serangkaian pola-pola input beserta pasangan pola-pola output yang menjadi targetnya. Dengan pengertian lain bahwa perlu adanya pola-pola referensi bagi JST. Hal ini penting sehingga setiap JST mengeluarkan output, ia akan membandingkan dengan hasil yang diharapkan tadi. Propagasi balik *error* inilah yang yang memberi nama JST sebagai JST BP.

Algoritma:

L0. Inisialisasi bobot (tetapkan dengan nilai acak kecil)
L1. Selama syarat kondisi *false* kerjakan langkah 2-9
L2. Untuk setiap pasangan yang akan dilakukan pembelajaran, kerjakan langkah 3-8

Umpan maju

L3. Setiap unit input (x_i, i=1,2,...,n) menerima sinyal input xi dan meneruskan ke semua unit dalam lapis tersembunyi

L4. Setiap unit tersembunyi (zj, j=1,2,...,p) menjumlahkan sampel input terbobotnya,

$$z\_in_j = v_{0j} + \sum_{i=1}^{n} x_i v_{ij}$$

Hitung sinyal output dengan fungsi aktivasinya

$$z_j = f(z\_in_j)$$

Kirimkan sinyal ini ke semua unit pada lapisan output.

L5. Setiap unit output (y_k, k=1,2,...,m) menjumlahkan sinyal input terbobotnya

$$y\_in_k = w_{0k} + \sum_{j=1}^{p} z_j w_{jk}$$

Hitung sinyal output dengan fungsi aktivasinya

$$y_k = f(y\_in_k)$$

Perambatan balik

L6. Setiap unit output (y_k, *k=1, .., m*) menerima pola target yang sesui dengan pola input pembelajaran, hitung suku informasi *error* nya

$$\delta_k = (t_k - y_k) f'(y\_in_k)$$

Hitung suku koreksi bobot (digunakan untuk perbaruan w_{jk}),

$$\Delta w_{jk} = \alpha \delta_k z_j$$

Hitung suku koreksi bias (digunakan untuk perbaruan w_{ok}),

$$\Delta w_{ok} = \alpha \delta_k$$

Kirimkan δ_k ke unit-unit dilapis bawahnya

L7. Setiap unit tersembunyi (z_j , *j =1, …, p*) menjumlahkan delta inputnya (dari unit-unit yang berada pada lapis diatasnya)

$$\delta\_in_j = \sum_{k=1}^{m} \delta_k w_{jk}$$

Hitung suku informasi *error*,

$$\delta_j = \delta\_in_j f'(z\_in_j)$$

Hitung suku koreksi bobot (untuk perbaruan v_{ij})

$$\Delta v_{ij} = \alpha \delta_j x_i$$

Hitung suku koreksi bias (untuk perbaruan v_{oj}),

$$\Delta v_{oj} = \alpha \delta_j$$

Perbaruan bobot dan bias:

L8. Setiap unit output (y_k, *k=1, .., m*) perbarui bobot-bobot dan bias nya:

$$w_{jk}(baru) = w_{jk}(lama) + \Delta w_{jk}$$

Setiap unit tersembunyi (z_j, *j=1,… , p*) perbarui bobot-bobot dan bias nya (*i=0,… , n*):

$$v_{ij}(baru) = v_{ij}(lama) + \Delta v_{ij}$$

L9. Uji syarat berhenti

Catatan:
Fungsi aktivasi sigmoid

$$f(x) = \frac{1}{1+e^{-x}}$$

$$f'(x) = f(x)\left[1 - f(x)\right]$$

B. Langkah Praktikum

1. Lakukanlah pelatihan dan pengujian fungsi XOR dengan laju pembelajaran α = 0.05 dan iterasi 500. Buat dan jalankan **Prak91a.m** (menggunakan fungsi toolbox NN)

```
%Prak91a.m
%Data pelatihan
%--------------------
```

```
pm1=[0 0]';
pm2=[0 1]';
pm3=[1 0]';
pm4=[1 1]';
pm=[pm1 pm2 pm3 pm4]
pt=[0 1 1 0]
%mode training
net=newff(minmax(pm),[2,1],{'tansig','tansig'},'traingd');
net.trainParam.show = 50;
net.trainParam.lr = 0.5;
net.trainParam.epochs = 500;
net.trainParam.goal = 1e-5;
[net,tr]=train(net,pm,pt);

%mode testing
p=input('Masukkan pola : ')
[yin]= sim(net,p)
```

2. Contoh sistem pengenalan wajah

```
%Prak92.m
%-------------
clc
clear
% Membaca data wajah untuk pelatihan terdiri dari 5 wajah orang yang berbeda
% dengan masing-masing menggunakan 4 sampel data
for i = 1:4
    data1{i} = imread(['C:\Program Files\MATLAB704\work\face\f1\' num2str(i) '.tif']);
    data2{i} = imread(['C:\Program Files\MATLAB704\work\face\f2\' num2str(i) '.tif']);
    data3{i} = imread(['C:\Program Files\MATLAB704\work\face\f3\' num2str(i) '.tif']);
    data4{i} = imread(['C:\Program Files\MATLAB704\work\face\f4\' num2str(i) '.tif']);
    data5{i} = imread([' C:\Program Files\MATLAB704\work\face\f5\' num2str(i) '.tif']);
end;

% Pemrosesan awal dan ekstraksi ciri
for i = 1:4
    data1{i} = imresize(data1{i},[30 20]);
    x1{i}=reshape(data1{i},[600,1]);
    x1{i}=x1{i}-mean(x1{i});
    x1{i}=x1{i}/max(x1{i});

    data2{i} = imresize(data2{i},[30 20]);
    x2{i}=reshape(data2{i},[600,1]);
    x2{i}=x2{i}-mean(x2{i});
    x2{i}=x2{i}/max(x2{i});

    data3{i} = imresize(data3{i},[30 20]);
    x3{i}=reshape(data3{i},[600,1]);
    x3{i}=x3{i}-mean(x3{i});
```

```
    x3{i}=x3{i}/max(x3{i});

    data4{i} = imresize(data4{i},[30 20]);
    x4{i}=reshape(data4{i},[600,1]);
    x4{i}=x4{i}-mean(x4{i});
    x4{i}=x4{i}/max(x4{i});

    data5{i} = imresize(data5{i},[30 20]);
    x5{i}=reshape(data5{i},[600,1]);
    x5{i}=x5{i}-mean(x5{i});
    x5{i}=x5{i}/max(x5{i});
end;

%F1
pm11=double(x1{1});
pm12=double(x1{2});
pm13=double(x1{3});
pm14=double(x1{4});

%F2
pm21=double(x2{1});
pm22=double(x2{2});
pm23=double(x2{3});
pm24=double(x2{4});

%F3
pm31=double(x3{1});
pm32=double(x3{2});
pm33=double(x3{3});
pm34=double(x3{4});

%F4
pm41=double(x4{1});
pm42=double(x4{2});
pm43=double(x4{3});
pm44=double(x4{4});

%F5
pm51=double(x5{1});
pm52=double(x5{2});
pm53=double(x5{3});
pm54=double(x5{4});

pm=[pm11 pm12 pm13 pm14 pm21 pm22 pm23 pm24 pm31 pm32 pm33 pm34 pm41
pm42 pm43 pm44 pm51 pm52 pm53 pm54];

 pt1=[0 0 0 0 1]';
 pt2=[0 0 0 1 0]';
 pt3=[0 0 1 0 0]';
```

```
pt4=[0 1 0 0 0]';
pt5=[1 0 0 0 0]';

pt=[pt1 pt1 pt1 pt1 pt2 pt2 pt2 pt2 pt3 pt3 pt3 pt3 pt4 pt4 pt4 pt4 pt5 pt5 pt5 pt5];

% jumlah neuron pada lapis tersembunyi 300
net=newff(minmax(pm),[300,5],{'tansig','tansig'},'traingd');
net.trainParam.show = 50;
net.trainParam.lr = 0.5;
net.trainParam.epochs = 500;
net.trainParam.goal = 1e-6;
[net,tr]=train(net,pm,pt);

%testing
Im = imread('C:\Program Files\MATLAB704\work\face\f1\5.tif');
I = imresize(Im,[30 20]);
x=reshape(I,[600,1]);
x=double(x-mean(x));
p=x/max(x);

[yin]= sim(net,p)
```

C. Evaluasi

1. Lakukan proses pelatihan dan pengujian dengan variasi parameter:
   a. Jumlah neuron pada lapis tersembunyi 300
   b. Laju pembelajaran 0,5
   c. Epoch: 500 dan 1000

Hasil Percobaan 1:

Masukan	Jumlah neuron lapis tersembunyi : 300 Epoch:500, laju pembelajaran: 0,5				
	F1	F2	F3	F4	F5
F1					
F2					
F3					
F4					
F5					
Akurasi (%)					

Hasil Percobaan 2:

Masukan	Jumlah neuron lapis tersembunyi : 300 Epoch:1000, laju pembelajaran: 0,5				
	F1	F2	F3	F4	F5
F1					
F2					
F3					
F4					
F5					
Akurasi (%)					

Nilai	Yogyakarta, ... Paraf asisten <.....>

ALGORITMA LVQ UNTUK PENGENALAN POLA

Pertemuan ke : X
Alokasi Waktu : 1,5 Jam
Kompetensi Dasar : Mahasiswa mampu memahami algoritma LVQ

Indikator : Memahami konsep algoritma LVQ untuk pengenalan pola

A. Teori Pendukung

Metode pembelajaran ini mengatur batas-batas antara kategori-kategori yang berbeda untuk menjaga keslahan klasifikasi seminimal mungkin. Pembelajaran dilakukan pada lapis kompetitif yang terawasi. Suatu lapis kompetitif akan secara otomatis belajar untuk mengklasifikasi vektor-vektor input. Klas-klas yang didapatkan sebagai hasil dari lapisan kompetitif ini hanya tergantung pada jarak antara vektor-vektor input.

Algoritma:

x : vektor pelatihan (x_1,..., x_i,..., x_n)
T : klas atau kategori vektor pembelajaran
$\mathbf{w_j}$: vektor bobot unit output; (w_{1j}, ..., w_{ij}, ..., w_{nj})
$\mathbf{C_j}$: kategori atau klas yang berikan oleh unit output j
$\|x-w_j\|$: jarak euclidean antara vektor input dan vektor

L0. Inisialisasi vektor referensi
Inisialisasi laju pembelajaran α(0)
L1. Selama syarat berhenti *false* kerjakan langkah 2-6
L2. Untuk setiap vektor input pembelajaran x, kerjakan langkah 3-4
L3. Cari J sehingga $\|x-w_j\|$ minimum
L4. Perbarui w_j:
Bila **T =C$_j$** , maka

$$w_j(baru) = w_j(lama) + \alpha.\left[x - w_j(lama)\right]$$

Bila **T ≠ C$_j$** , maka

$$w_j(baru) = w_j(lama) - \alpha.\left[x - w_j(lama)\right]$$

L5. Kurangi laju pembelajaran
L6. Uji syarat berhenti;
Syaratnya: spesifikasi cacah iterasi atau laju pembelajaran mencapai nilai yang cukup kecil.

B. Langkah Praktikum

1. Jalankan program berikut:

```
%Prak10.m
p1=[1;1;0;0];
p2=[0;0;0;1];
p3=[0;0;1;1];
p4=[1;0;0;0];
```

```
p5=[0;1;1;0];

P = [p1 p2 p3 p4 p5];
C = [1 2 1 2 2];
T = ind2vec(C);

%mode training
net = newlvq(minmax(P),4,[0.6 0.4],0.01);
net=train(net,P,T);

%mode testing
p = input('Masukkan pola baru :')

a = vec2ind(sim(net,p))
```

2. Implementasikan sistem pengenalan wajah pada Praktikum 9 dengan menggunakan algoritma LVQ

Hasil percobaan 1:

Masukan	Jumlah neuron lapis tersembunyi : 300				
	F1	F2	F3	F4	F5
F1					
F2					
F3					
F4					
F5					
Akurasi (%)					

Hasil percobaan 2:

Masukan	Jumlah neuron lapis tersembunyi : 600				
	F1	F2	F3	F4	F5
F1					
F2					
F3					
F4					
F5					
Akurasi (%)					

C. Evaluasi

Nilai	Yogyakarta, ... Paraf asisten <.....>

Referensi:

1. Tou, Julius T and Gonzalez, Rafael C., *Pattern Recogniton Principles*, Addison-Wesley Publishing Company, London
2. Schalkoff, R.J., 1992, *Pattern Recognition: Statistical, Structural and Neural Approaches,* John Wiley & Sons, Inc., Singapore
3. Duda R.O., Hart P.E., Stork D.G., 2000, *Pattern classification,* 2ed., Wiley, ISBN 0471056693
4. Fausett, L., 1994, *Fundamentals of Neural Networks: architecture, algoritma, and applications*, Prentice Hall, New Jersey.
5. Sivanandam,S.N., Sumathi, S., and Deepa, S.N., 2006, ***Introduction to Neural Networks using Matlab 6.0***, The McGraw-Hill Comp.